ICASE

A TRUST REGION ALGORITHM FOR EQUALITY CONSTRAINED MINIMIZATION:

CONVERGENCE PROPERTIES AND IMPLEMENTATION

(NASA-CR-185789) A TRUST REGION ALGORITHM FOR EQUALITY CONSTRAINED MINIMIZATION: CONVERGENCE PROPERTIES AND IMPLEMENTATION (ICASE) 35 p

N89-71427

Unclas

00/61 0224382

Avi Vardi

Report No. 81-9

February 20, 1981

INSTITUTE FOR COMPUTER APPLICATIONS IN SCIENCE AND ENGINEERING
NASA Langley Research Center, Hampton, Virginia

Operated by the

UNIVERSITIES SPACE USRA RESEARCH ASSOCIATION

A TRUST REGION ALGORITHM FOR EQUALITY CONSTRAINED MINIMIZATION: CONVERGENCE PROPERTIES AND IMPLEMENTATION

Avi Vardi

Institute for Computer Applications in Science and Engineering

ABSTRACT

In unconstrained minimization, trust region algorithms use directions that are a combination of the quasi-Newton direction and the steepest descent direction, depending on the fit between the quadratic approximation of the function and the function itself.

Algorithms for nonlinear constrained minimization problems usually determine a quasi-Newton direction and use a line search technique to determine the step. Since trust region strategies have proved to be successful in unconstrained minimization, we develop a new trust region strategy for equality constrained minimization. This algorithm is analyzed and global as well as local superlinear convergence theorems are proved for various versions.

We demonstrate how to implement this algorithm in a numerically stable way. A computer program based on this algorithm has performed very satisfactorily on test problems; numerical results are provided.

This research was partially supported under NSF Grants No. ECS-7921279 and No. ENG 76-08749, and under NASA Contract No. NAS1-15810 while the author was in residence at ICASE, NASA Langley Research Center, Hampton, VA 23665.

I. Introduction

Consider the problem of minimizing a smooth nonlinear function subject to nonlinear constraints: $\min_x f(x)$ subject to $h_j(x) = 0$, $i = 1,\ldots m$; $m \leq n$. Most of the methods for this problem attempt to transform the constrained problem to a related unconstrained problem of minimization or solving equations and then to translate the quasi-Newton technique on the unconstrained problem back to the constrained problem. We will mention a few quasi-Newton methods that are related to this work. They deal with one type of algorithm: at each iteration a quasi-Newton step is generated and then a line search technique is used to decide which point should be accepted. For the purpose of deciding which point to accept, a penalty function is designed. This function may look like $PL(x) = f(x) + \Sigma\mu_i |h_i(x)|$ where the μ_i's are penalty parameters that have to be determined.

Han [16], using the quadratic programming approach without line searches, and Tapia [35] using the multipliers update formula approach generalize the local convergence theory of Broyden, et al. [3] and prove q-superlinear convergence. One paper that considers the effect of the penalty function on local convergence is Chamberlain, et al. [4].

In all these approaches the algorithms use approximations to the Hessian of the Lagrangian function with respect to x. In order to guarantee local q-superlinear convergence we must assume that the initial Hessian approximation is close enough to the Hessian at the solution which may not be positive definite. Powell [24] analyzes local convergence and explains why positive definite Hessian approximations can still be used.

Han [14], under some conditions that bound the Hessian approximations, establishes global convergence when exact line search (for a penalty function) is used. A similar analysis with more emphasis on implementation appears in Powell [25].

In section 2 we derive the quasi-Newton step for the problem. The main contribution of this paper is the introduction of a trust region algorithm for equality constrained minimization and this is the subject of section 3. The idea behind the algorithm is that at each iteration the quadratic approximation to the function which is used to obtain the quasi-Newton step is trusted only within a sphere of radius r around the current point. This strategy has proved to be successful in unconstrained minimization (see [31]). The convergence properties of the algorithm are presented in section 4 and in section 5 we demonstrate how to implement the algorithm and provide numerical results.

Notation Convention: We use superscripts to denote the iteration we are in. Thus Δx^k is the step at the k^{th} iteration. In order to avoid confusion with powers of matrices we use parentheses in the following way: $(B^k)^j$ denotes the j^{th} power of the matrix B^k where B^k is a matrix associated with the k^{th} iteration.

We also often replace for convenience $\binom{x}{v}$ by (x,v) and $\binom{\Delta x}{\Delta v}$ by $(\Delta x,\Delta v)$. All vector norms are Euclidean norms, i.e., $\| x\| = \| x\|_2 = (\Sigma x_i^2)^{\frac{1}{2}}$.

2. A Quasi-Newton Algorithm

Consider $m+1$ real valued functions $f,h_1,\ldots,h_m$ defined on $\mathbb{R}^n$. We are interested in solving the problem

$$\min_{x:h(x)=0} \quad f(x). \tag{2.1}$$

We will actually try to find a local minimizer for this problem, i.e., a feasible point x^* such that there exists a $\delta > 0$ such that for all x satisfying $h(x) = 0$ and $\| x - x^*\| < \delta$, $f(x^*) \leq f(x)$. Assign Lagrange multipliers $v_1,\ldots,v_m$ to each of the constraints and form the Lagrangian function

(2.2) $$L(x,v) = f(x) + h(x)^T v \quad ,$$

The gradient of L will be denoted by

$$\nabla L(x,y) = \begin{pmatrix} \nabla_x L(x,v) \\ \nabla_v L(x,v) \end{pmatrix} = \begin{pmatrix} \nabla f(x) + \nabla h(x) v \\ h(x) \end{pmatrix} .$$

where

$$\nabla f(x) = \begin{pmatrix} \frac{\partial f}{\partial x_1}(x) \\ \vdots \\ \frac{\partial f}{\partial x_n}(x) \end{pmatrix} \quad \text{and} \quad \nabla h(x) = \begin{pmatrix} \frac{\partial h_1}{\partial x_1}(x) & \cdots & \frac{\partial h_m}{\partial x_1}(x) \\ \vdots & & \vdots \\ \frac{\partial h_1}{\partial x_n}(x) & \cdots & \frac{\partial h_m}{\partial x_n}(x) \end{pmatrix} .$$

The Hessian matrix of L will be denoted by

(2.3)

$$\nabla^2 L(x,v) = \begin{pmatrix} \nabla^2_{xx} L(x,v) & \nabla^2_{xv} L(x,v) \\ \nabla^2_{vx} L(x,v) & \nabla^2_{vv} L(x,v) \end{pmatrix} = \begin{pmatrix} \nabla^2 f(x) + \Sigma v_i \nabla^2 h_i(x) & \nabla h(x) \\ \nabla h(x)^T & 0 \end{pmatrix} ,$$

where

$$\nabla^2 f(x) = \left[\frac{\partial^2 f}{\partial x_k \partial x_\ell} (x) \right]_{k,\ell=1,\dots,n} ,$$

and

$$\nabla^2 h_i(x) = \left[\frac{\partial^2 h_i}{\partial x_k \partial x_\ell} (x) \right]_{k,\ell=1,\dots,n} \quad \text{for } i = 1,\dots,m \quad .$$

We will assume that $f, h_1, \ldots, h_m$ are twice continuously differentiable and there exists a Lipschitz constant K such that

$$\| \nabla^2 f(x) - \nabla^2 f(y) \| \leq K \| x - y \| ,$$

and

$$\| \nabla^2 h_i(x) - \nabla^2 h_i(y) \| \leq K \| x - y \| \ \forall\, i = 1, \ldots, m \qquad \forall\, x,y \in \mathbb{R}^n ,$$

and that f is bounded below.

At a local minimizer x^* there exists $v^* \in \mathbb{R}^m$ such that $\nabla L(x^*,V^*) = 0$ and for all $z \in \mathbb{R}^m$ such that $\nabla h(x^*)^T z = 0$, $z^T \nabla^2_{xx} L(x^*,V^*) z \geq 0$. We assume that $\nabla h(x^*)$ is of full rank and that for all $z \in \mathbb{R}^m$, $\nabla h(x^*)^T z = 0$, $z^T \nabla^2_{xx} L(x^*,V^*) z > 0$. We also assume, for now, that $\nabla h(x)$ is of full rank for all $x \in \mathbb{R}^n$; we will show in section 5 how to handle the case when $\nabla h(x)$ is not of full rank.

Looking at the Hessian matrix $\nabla^2 L(x,v)$ as defined by (2.3) we see that the term $\nabla^2_{xx} L(x,v)$ requires second derivatives of f and h_i's. Since in practice second derivatives are often unavailable we use approximations $B \approx \nabla^2_{xx} L(x,v)$. Having the matrix B we also obtain an approximation to the matrix $\nabla^2 L(x,v)$ of the form

$$\overline{B} = \begin{pmatrix} B & \nabla h(x) \\ \nabla h(x)^T & 0 \end{pmatrix} \approx \nabla^2 L(x,v) = \begin{pmatrix} \nabla^2_{xx} L(x,v) & \nabla h(x) \\ \nabla h(x)^T & 0 \end{pmatrix} .$$

When B is replaced by $B^k \approx \nabla^2_{xx} L(x^k,v^k)$ the matrix will be used to obtain the step $\Delta x^k, \Delta v^k$ and hence $x^{k+1} = x^k + \Delta x^k$ and $v^{k+1} = v^k + \Delta v^k$. The means for obtaining Δx^k, Δv^k are described in this section and section 3. In order to continue the iterative process B^k must be updated to B^{k+1}. We use the BFGS update formula to update B^k at each iteration. The BFGS update is due to Broyden [2], Fletcher [10], Goldfarb [12], and Shanno [26]. If . If $y^k = \nabla_x L(x^{k+1}, v^{k+1}) - \nabla_x L(x^k, v^{k+1})$ and $(y^k)^T \nabla x^k > 0$ then

(2.4) $$B^{k+1} = B^k + \frac{y^k y^{k^T}}{y^{k^T} \Delta x^k} - \frac{B^k \Delta x^k \Delta x^{k^T} B^k}{\Delta x^{k^T} B^k \Delta x^k} .$$

Otherwise $B^{k+1} = B^k$. If B^k is symmetric and positive definite then so is B^{k+1}. Moreover if B^{k+1} is given by (2.4) it satisfies the secant equation $B^{k+1} x^k = y^k$. The matrix $\nabla^2_{xx} L(x^*, v^*)$ is not necessarily positive definite but the use of positive definite update guarantees, as will be seen later, descent directions and will not interfere with fast local convergence. In the implementation we will store and work with the Cholesky decomposition of the matrix, $B^k = C^k C^{k^T}$.

We can now present the quasi-Newton (Q.N.) step for this problem:

(2.5) $$\begin{pmatrix} \Delta x^k \\ \Delta v^k \end{pmatrix} = -(\bar{B}^k)^{-1} \nabla L(x^k, v^k) = -\begin{pmatrix} B^k & \nabla h(x^k) \\ \nabla h(x^k)^T & 0 \end{pmatrix}^{-1} \begin{pmatrix} \nabla f(x^k) + \nabla h(x^k) v^k \\ h(x^k) \end{pmatrix} .$$

Quasi-Newton Algorithms usually employ a line search to determine an $0 \le \alpha^k \le 1$ such that $x^{k+1} = x^k + \alpha^k \Delta x^k$, $v^{k+1} = v^k + \alpha^k \Delta v^k$ satisfy a certain criterion (like a decrease of a Penalty function).

An equivalent way of computing the Q. N. step is to solve the quadratic programming problem

$$(2.6)\quad \begin{cases} \min\limits_{\Delta x} L(x^k,v^k) + \nabla_x L(x^k v^k)^T \Delta x + \frac{1}{2}\Delta x^T B^k \Delta x\ , \\ \text{s.t.} \\ h(x^k) + \nabla h(x^k)^T \Delta x = 0\ . \end{cases}$$

Now Δx^k is the solution of (2.6) and Δv^k is the vector of Lagrange multipliers corresponding to the linear constraints. The objective function in (2.6) may be looked at as a quadratic approximation of the Lagrangian function with Lagrange multipler v^k. The quadratic problem thus consists of minimizing this quadratic approximation subject to the original constraints, linearized at the point x^k.

3. The Trust Region Algorithm

We first motivate our method and then provide the details. In a Trust Region algorithm we have at each iteration a Trust Region sphere around the current point $\{x^k + \Delta x\colon \|\Delta x\| \le r^k\}$ in which we trust the quadratic approximation of the function or in our case the quadratic approximation to the Lagrangian function as in (2.6). (The determination of r^k will be discussed later in this section.) An attempt to add a trust region to (2.6) results in the problem

$$(3.1)\quad \begin{cases} \min\limits_{\Delta x} L(x^k,v^k) + \nabla_x L(x^k v^k)^T \Delta x + \frac{1}{2}\Delta x^T B^k \Delta x, \\ \text{s.t.} \\ h(x^k) + \nabla h(x^k)^T \Delta x = 0 \\ \|\Delta x\| \le r^k\ , \end{cases}$$

and we can immediately observe a difficulty: If r^k is too small we may have $\{\Delta x : h(x^k) + \nabla h(x^k)^T \Delta x = 0\} \cap \{\Delta x : \| \Delta x \| \le r^k\} = \phi$. Thus even the shortest step Δx that satisfies $h(x^k) + \nabla h(x^k)^T \Delta x = 0$ lies outside the Trust Region. We suggest therefore to change the linear constraint in (3.1) into $\alpha h(x^k) + \nabla h(x^k)^T \Delta x = 0$ where α depends on the radius r^k and is determined such that $\{\Delta x : \alpha h(x^k) + \nabla h(x^k)^T \Delta x = 0\} \cap \{\Delta x : \| \Delta x \| \le r^k\} \neq \phi$.

We now give more details on our method. In order to simplify the notations we omit the superscripts. Assume we are at the current point x with Lagrange multiplier v. We take a Q-R decomposition of $\nabla h(x)$ with column pivoting, i.e., find an orthogonal matrix Q, a permutation matrix Π and a nonsingular upper triangular $m \times m$ matrix T such that $Q \nabla h(x) \Pi = \binom{T}{0}$. Recall that we have assumed in this section that $\nabla h(x)$ is of full rank for all $x \in \mathbb{R}^n$. We partition Q by $Q = \begin{pmatrix} \overline{Q} \\ \overline{\overline{Q}} \end{pmatrix}$, where $\overline{Q}$ has m rows.

Let $\Delta x(\lambda), \Delta v(\lambda)$ be computed in the following form:

$$(3.2) \qquad \begin{bmatrix} \Delta x(\lambda) \\ \Delta v(\lambda) \end{bmatrix} = - \begin{bmatrix} B + \lambda I & \nabla h(x) \\ \nabla h(x)^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} \nabla_x L(x,v) \\ \alpha(\lambda) h(x) \end{bmatrix},$$

where

$$\alpha(\lambda) = \min\left\{1, \max\left\{\frac{1}{\lambda}, \frac{1}{\lambda^2} \frac{\|B\| \; \|\overline{\overline{Q}} \nabla f(x)\|}{\|T^{-T} \Pi^T h(x)\|}\right\}\right\}.$$

and λ chosen so that $\|\Delta x(\lambda)\| \le r$. Theorem 3.2 summarizes the characteristics of $\Delta x(\lambda)$, $\Delta v(\lambda)$ as a function of λ in the new model. We first need the following lemma:

<u>Lemma 3.1</u> For any integer $p > 0$, matrix B and $g \subset \mathbb{R}^n$, we have that $\lambda^p \cdot (B + \lambda I)^{-p} \cdot g \to g$ as $\lambda \to \infty$.

<u>Theorem 3.2</u> Consider $(\Delta x(\lambda), \Delta v(\lambda))$ as defined in (3.2). Then $\Delta x(\lambda)$ and $(v + \Delta v(\lambda))$ do not depend on v for all $\lambda \geq 0$. When λ tends to infinity, $v + \Delta v(\lambda) \to \tilde{v}_+ \equiv [\nabla h(x)^T \nabla h(x)]^{-1}[h(x) - \nabla h(x)^T \nabla f(x)]$ and $\Delta x(\lambda) - \frac{1}{\lambda} \nabla L(x, \tilde{v}_+) \to 0$. Finally, $\| \Delta x(\lambda) \|$ is a monotonically decreasing function of $\lambda \geq 0$.

<u>Proof</u>: By multiplying with $\begin{pmatrix} B + \lambda I & \nabla h(x) \\ \nabla h(x)^T & 0 \end{pmatrix}$ on both sides in (3.2) we obtain

$$(3.3) \qquad (B + \lambda I)\Delta x + \nabla h(x)\ \Delta v + \nabla_x L(x,v) = 0,$$

$$(3.4) \qquad \nabla h(x)^T \Delta x + \alpha(\lambda) h(x) = 0.$$

From (3.3) we get

$$(3.5) \qquad \Delta x = -(B + \lambda I)^{-1} \nabla_x L(x, v + \Delta v).$$

From (3.4) and (3.5) we have

$$(3.6) \quad \Delta v = [\nabla h(x)^T (B + \lambda I)^{-1} \nabla h(x)]^{-1} [\alpha(\lambda) \cdot h(x) - \nabla h(x)^T (B + \lambda I)^{-1} \nabla_x L(x,v)],$$

which implies that

$$(3.7) \quad v + \Delta v = [\nabla h(x)^T (B + \lambda I)^{-1} \nabla h(x)]^{-1} [\alpha(\lambda) \cdot h(x) - \nabla h(x)^T (B + \lambda I)^{-1} \nabla f(x)].$$

Thus $\Delta x(\lambda)$ and $(v+\Delta v(\lambda))$ do not depend on v for all $\lambda \geq 0$. To get the asymptotic behavior of $\Delta x(\lambda)$, $\Delta v(\lambda)$ we use lemma 3.1. Note that when λ is large, $\alpha(\lambda) = 1/\lambda$. Thus when λ tends to infinity

$$v+\Delta v(\lambda) = [\lambda \nabla h(x)^T (B+\lambda I)^{-1} \nabla h(x)]^{-1} [h(x) - \lambda \nabla h(x)^T (B+\lambda I)^{-1} \nabla f(x)]$$

$$\rightarrow [\nabla h(x)^T \nabla h(x)]^{-1} [h(x) - \nabla h(x)^T \nabla f(x)] = \tilde{v}_+ \quad .$$

and from (3.5)

$$\Delta x(\lambda) \approx -\frac{1}{\lambda} \nabla_x L(x, \tilde{v}_+) \quad .$$

The proof that $\|\Delta x\|$ is monotonically decreasing to zero is straightforward but somewhat long and is omitted here. For a proof see [31].

One of the major difficulties in constrained optimization is in deciding whether (x^{k+1}, v^{k+1}) is a better approximation to the solution than (x^k, v^k). The Lagrangian function $L(x,v) = f(x) + h(x)^T v$ cannot serve for this purpose because of the risk of cycling. Instead we use the penalty function

$$PL(x) = f(x) + \Sigma \mu_i |h_i(x)| , \tag{3.8}$$

where the μ_i's are positive fixed constants that are called penalty coefficients.

Before stating the algorithm it is important to show that at each iteration there will be a (radius small enough or) $\lambda > 0$ large enough such that a penalty function of the form of (3.8) (which will replace the Lagrangian in the algorithm in order to achieve global convergence) is also decreased.

<u>Theorem 3.3</u> If $(\Delta x, \Delta v)$ is computed as in (3.2) and $\nabla L(x,\tilde{v}_+) \neq 0$, then there exists $\lambda > 0$ large enough such that $L(x+\Delta x(\lambda), v+\Delta v(\lambda)) < L(x, v+\Delta v(\lambda))$. Further assume that $\forall\ i = 1,\dots,m$

$$\mu_i > |\tilde{v}_{+i}| = |[(\nabla h(x)^T \nabla h(x))^{-1}(h(x) - h(x)^T \nabla f(x))]_i| \tag{3.9}$$

Then there exists $\lambda > 0$ large enough such that $PL(x+\Delta x(\lambda)) < PL(x)$ where $PL(x) = f(x) + \Sigma\mu_i|h_i(x)|$.

<u>Proof</u>. We will use the asymptotic behavior of $\Delta x, \Delta v$ as established by Theorem 3.2. From Taylor's theorem we have

$$\lambda[L(x+\Delta x, v+\Delta v) - L(x, v+\Delta v)] = \lambda\nabla_x L(x+\nu\Delta x, v+\Delta v)^T \Delta x \quad \text{for some} \quad \nu \in (0.1).$$

We now use Lemma 3.1 and the fact that $\nabla L(x,\tilde{v}_+) \neq 0$ implies $\nabla_x L(x,\tilde{v}_+) \neq 0$ to obtain that when λ tends to infinity

$$\lambda[L(x+\Delta x, v+\Delta v) - L(x, v+\Delta v)] \to -\nabla_x L(x,\tilde{v}_+)^T \nabla_x L(x,\tilde{v}_+) < 0.$$

Next consider $PL(x+\Delta x)$:

$$\begin{aligned}
\lambda[PL(x+\Delta x) - PL(x)] &= \\
&= \lambda[f(x+\Delta x) - f(x) + \Sigma\mu_i(|h_i(x+\Delta x)| - |h_i(x)|)] = \\
&= \lambda[\nabla f(x)^T\Delta x + \tfrac{1}{2}\Delta x^T \nabla^2 f(x+\nu\Delta x)\Delta x \\
&\quad + \sum_{i:h_i(x)\neq 0} \mu_i \operatorname{sign}(h_i(x))(\nabla h_i(x)^T\Delta x + \tfrac{1}{2}\Delta x^T\nabla^2 h_i(x+\nu\Delta x)\Delta x) \\
&\quad + \sum_{i:h_i(x)=0} \mu_i\sigma_i(\nabla h_i(x)^T\Delta x + \tfrac{1}{2}\Delta x^T\nabla^2 h_i(x+\nu\Delta x)\Delta x)].
\end{aligned}$$

for some $\nu \in [0,1]$ and $\sigma_i = \pm 1$. The sign of σ_i is not important because as we shall see, the third term tends to zero when λ tends to infinity. Thus

$$\begin{aligned}\lambda[PL(x+\Delta x) - PL(x)] = \lambda[&\nabla f(x)^T \Delta x + \tfrac{1}{2}\Delta x^T \nabla^2 f(x+\nu\Delta x)\Delta x \\ &+ \sum_{i:h_i(x)\neq 0} \mu_i \operatorname{sign}(h_i(x))(-\alpha(\lambda)h_i(x) + \tfrac{1}{2}\Delta x^T \nabla^2 h_i(x+\nu\Delta x)\Delta x) \\ &+ \sum_{i:h_i(x)=0} \mu_i \sigma_i (\tfrac{1}{2}\Delta x^T \nabla^2 h_i(x+\nu\Delta x)\Delta x)].\end{aligned}$$

When λ tends to infinity, $\lambda\Delta x \to -\nabla_x L(x,\tilde{v}_+)$ and $\lambda\cdot\alpha(\lambda) = 1$. Thus

$$\begin{aligned}\lambda[PL(x+\Delta x) - PL(x)] &\to \nabla f(x)^T \nabla_x L(x,\tilde{v}_+) - \sum_{i:h_i(x)\neq 0} \mu_i \operatorname{sign}(h_i(x))h_i(x) \\ &= -\nabla_x L(x,\tilde{v}_+)^T \nabla_x L(x,\tilde{v}_+) - \sum_{i:h_i(x)\neq 0} (\mu_i \operatorname{sign}(h_i(x)) - \tilde{v}_{+i})h_i(x),\end{aligned}$$

and because the choice of μ_i's in (3.9) is such that $\mu_i > |\tilde{v}_{+i}|$ for all $i=1,\dots,m$ the last expression is negative and this completes the proof of the theorem.

We would like to emphasize two nice properties of this model:

a. When r is large enough, the full Q.N. step is taken and $\lambda = 0$. When r is very small and as a consequence λ is very large, $\Delta x \approx -\frac{1}{\lambda}\nabla_x L(x,\tilde{v}_+)$ is a steepest descent direction in minimizing $L(x,\tilde{v}_+)$. $\tilde{v}_+$ as defined in Theorem 3.2 does not depend on B and has been used in other algorithms for equality constrained optimization. (See, e.g. [30].)

b. If a bound on the right-hand side in (3.9) is known, then we can fix the parameters μ_i in the penalty function and there is no need to change the μ_i's in the process of the convergence with the risk of cycling.

Here is our algorithm.

<u>Algorithm I</u>

Step 1: Start with x^0, v^0, r^0, B^0, $k=-1$.

Step 2: $k = k+1$.

Step 3: Find λ^k (first try $\lambda^k = 0$) and Δx^k, Δv^k such that

$$(3.10)\quad \begin{cases} \begin{pmatrix} \Delta x^k \\ \Delta x^k \end{pmatrix} = \begin{pmatrix} \Delta x^k(\lambda^k) \\ \Delta v^k(\lambda^k) \end{pmatrix} = - \begin{pmatrix} B^k + \lambda^k I & \nabla h(x^k) \\ \nabla h(x^k)^T & 0 \end{pmatrix}^{-1} \begin{pmatrix} \nabla_x L(x^k,v^k) \\ \alpha(\lambda^k)h(x^k) \end{pmatrix} \\ \\ \| \Delta x^k \| \le r^k, \quad \lambda^k \ge 0 \quad \text{and} \quad \lambda^k(\| \Delta x^k \| - r^k) = 0. \end{cases}$$

Step 4: Check if $PL(x^k + \Delta x^k) < PL(x^k)$. If not, reduce r^k and return to Step 3.

(If a bound as in (3.9) is not known, μ_i may be chosen by a method ndescnibed in section 5c).

Step 5: Check for convergence. If not achieved, continue.

Step 6: Compare the quadratic approximation of L with the value of the Lagrangian function and its gradient at $(x^k + \Delta x^k, v^k + \Delta v^k)$ and accordingly increase or decrease r^{k+1}.

Step 7: Compute $y^k = \nabla_x L(x^k + \Delta x^k, v^k + \Delta v^k) - \nabla_x L(x^k, v^k + \Delta v^k)$ and update $B^{k+1} = \mathrm{BFGS}(B^k, \Delta x^k, y^k)$.

Step 8: $x^{k+1} = x^k + \Delta x^k$; $v^{k+1} = v^k + \Delta v^k$; return to Step 2.

4. Convergence Properties

The global convergence theorem that appears in this section assumes that the choice of λ^k in Step 3 of Algorithm I is not such that $\|(\Delta x^k,\Delta v^k)\| \leq r^k$, but instead at each iteration λ^k is chosen to minimize the penalty function $PL(x^k+\Delta x^k)$.

Thus we could restate Step 3 and 4 of the algorithm as follows:

Find λ^k, $\Delta x^k = \Delta x^k(\lambda^k)$ and $\Delta v^k(\lambda^k)$ such that

$$PL(x^k+\Delta x^k(\lambda^k)) = \min_{\lambda:\lambda\geq 0} PL(x^k+\Delta x^k(\lambda)). \tag{4.1}$$

(We will refer to this as "exact λ-search.")

The following theorem which gives the global properties of the algorithm is similar to Theorem 3.2 in [14].

Theorem 4.1. Assume Algorithm I is performed with exact λ-search (4.1) and there exist two positive numbers α,β such that $\alpha z^T z \leq z^T B^k z \leq \beta z^T z$ for each k and any $z \in \mathbb{R}^n$. Also assume that μ_i, $i = 1,\ldots,m$ satisfy inequality (3.9) at all points x^k generated by the algorithm and the set $\{x: f(x) + \Sigma\mu_i|h_i(x)| < f(x^0) + \Sigma\mu_i|h_i(x^0)|\}$ is bounded. Then the sequence $\{x^k\}$ remains bounded, and if $\tilde{x}$ is any accumulation point, $h(\tilde{x}) = 0$ and there exists a $v \in \mathbb{R}^m$ such that $\nabla f(\tilde{x}) + \nabla h(\tilde{x})\tilde{v} = 0$.

Proof. First observe that if $\Delta x^k = 0$, then from theorem 3.2 $\nabla_x L(x^k,v^k+\Delta v^k) = 0$ and $h(x^k) = 0$. That means that $(x^k,v^k+\Delta v^k)$ satisfies the required conditions. Suppose now that for all k, $\Delta x^k \neq 0$. From the algorithm we have for all k, $PL(x^k+\Delta x^k) < PL(x^k)$. Thus, because of the assumptions $\{x^k\}$ is bounded and has an accumulation point $\tilde{x}$. Without loss of generality we may assume (by taking a subsequence) that $x^k \to \tilde{x}$, $B^k \to \tilde{B}$. ($\tilde{B}$ exists and is positive definite because of the assumed existence of α,β.) Let $\Delta\tilde{x}(\lambda)$, $(\tilde{v}+\Delta v(\tilde{\lambda}))$ be determined by (3.2) with $\tilde{x}$, $\tilde{B}$ replacing x and B. (According to theorem 3.2, $\Delta\tilde{x}(\lambda)$ and $(\tilde{v}+\Delta\tilde{v}(\lambda))$ do not depend on $\tilde{v}$.) Let $\tilde{\lambda}$ be such that $PL(\tilde{x}+\Delta\tilde{x}(\lambda)) = \min_{\lambda:\lambda\geq 0} PL(\tilde{x}+\Delta\tilde{x}(\lambda))$. Let $\Delta\tilde{x} = \Delta\tilde{x}(\tilde{\lambda})$ and $(\tilde{v}+\Delta\tilde{v}) = (\tilde{v}+\Delta\tilde{v}(\tilde{\lambda}))$.

If $\Delta\tilde{x} = 0$, then as above we see that $\nabla L(\tilde{x}, \tilde{v}+\Delta\tilde{v}) = 0$. If $\Delta\tilde{x} \neq 0$, we will obtain a contradiction. Our assumptions imply that $\Delta x^k(\tilde{\lambda}) \to \Delta\tilde{x}$. From theorem 3.3 we can conclude that $PL(\tilde{x}+\Delta\tilde{x}) < PL(\tilde{x})$. Let $\beta = PL(\tilde{x}+\Delta\tilde{x}) - PL(\tilde{x})$. Since $x^k + \Delta x^k(\tilde{\lambda}) \to \tilde{x} + \Delta\tilde{x}(\tilde{\lambda})$ it follows that for a sufficiently large k

$$PL(x^k+\Delta x^k(\tilde{\lambda})) + \frac{\beta}{2} < PL(\tilde{x}). \tag{4.2}$$

But $PL(\tilde{x}) < PL(x^{k+1}) = \min_{\lambda:\lambda\geq 0} PL(x^k+\Delta x^k(\lambda)) < PL(x^k+\Delta x^k(\tilde{\lambda})) + \frac{\beta}{2}$, which for k large enough contradicts (4.2). This completes the proof of the theorem.

We now establish the local properties of the algorithm. We assume that $\lim \{(x^k,v^k)\} = (x^*,v^*)$, a local solution, and that $\Delta x^k \neq 0$ for all k. We will also assume, for now, that $\nabla^2_{xx}L(x^*,v^*)$ is a positive definite matrix. We will discuss the other case at the end of this section. Another problem is that Algorithm I uses the penalty function $PL(x)$ to decide in Step 4 whether to accept the step. This may interfere with fast local convergence. In [4] an example is given which shows that even when we are arbitrarily close to the solution, the use of the penalty function may prevent superlinear convergence. Our way of handling this problem is to change slightly the penalty function so that in a neighborhood of feasible points the regular Lagrangian function $L(x,v) = f(x) + h(x)^T v$

will replace the penalty function in Step 4. More details appear in section 5. We will still use the penalty function when the point is not in the neighborhood of a feasible point in order to retain the global properties of the algorithm.

Since the following discussion deals with what happens in a neighborhood of the solution, we will refer to algorithm II which represents an interpretation of algorithm I in which $L(x,v)$ is used instead of the penalty function.

<u>Algorithm II</u>

```
Step 1:  Start with $x^0$, $v^0$, $r^0$, $B^0$, $k = 1$.
Step 2:  $k = k+1$.
Step 3   Find $r^k$ such that if the step is computed according to (3.10)
 and 4:
(4.3)         $L(x^k+\Delta x^k, v^k+\Delta v^k) < L(x^k, v^k+\Delta v^k)$.
         (Try first $r^k = \| (\bar{B}^k)^{-1} \nabla L(x^k,v^k) \|$, i.e., check whether the
         Q.N. step satisfies condition (4.3).)
Step 5:  Check for convergence. If not achieved, continue.
Step 6:  Compute $y^k = \nabla_x L(x^k+\Delta x^k, v^k+\Delta v^k) - \nabla_x L(x^k, v^k+\Delta v^k)$ and update
         $B^{k+1} = BFGS(B^k, \Delta x^k, y^k)$.
Step 7:  $x^{k+1} = x^k + \Delta x^k$; $v^{k+1} = v^k + \Delta v^k$. Return to Step 2.
```

We will show that there exist $\varepsilon > 0$, $\delta > 0$ such that if $\| x^0 - x^* \| < \varepsilon$, $\| (B^0)^{-1} - \nabla^2_{xx} L(x^*,v^*)^{-1} \| < \delta$ (such a bound is equivalent to a bound on $\| B^0 - \nabla^2 L(x^*,v^*) \|$), then the convergence is q-superlinear. The main tools for local convergence in unconstrained problems were established in [3]. Lemma 4.2 is proved there; (we replace the unconstrained function with the Lagrangian function with a fixed Lagrange multiplier v^*.)

Let $M = [\nabla_{xx}^2 L(x^*,v^*)]^{\frac{1}{2}}$. Define the matrix norm $\|A\|_M = \|MAM\|_F$ where $\|D\|_F = \sqrt{\sum_{ij} d_{ij}^2}$.

We will also use the fact that there exist a constant $\eta > 0$ such that for every $n \times n$ matrix A

(4.4) $$\|A\|_F < \eta \|A\|_M .$$

Lemma 4.2 Let (x^*,v^*) be a local solution. For all $z^1, z^2 \in \mathbb{R}^n$

$$\|\nabla_x L(z^1,v^*) - \nabla_x L(z^2,v^*) - \nabla_{xx}^2 L(x^*,v^*)\| \leq K\sigma(z^1,z^2)\cdot\|z^2 - z^1\|$$

where $\sigma(z^1,z^2) = \max\{\|z^1 - x^*\|, \|z^2 - x^*\|\}$. Furthermore, if $\nabla_{xx}^2 L(x^*,v^*)$ is invertible there exist $\varepsilon > 0$ and $\rho > 0$ such that $\sigma(z^1,z^2) \leq \varepsilon$ implies that $\frac{1}{\rho}\|z^2 - z^1\| < \|\nabla_x L(z^2,v^*) - \nabla_x L(z^1,v^*)\| \leq \rho\|z^2 - z^1\|$.

Another important result in the paper by Broyden, Dennis, and More is the bounded deterioration of (the inverse of or) the Hessian approximations. Han [13] and Tapia [29] observed that this result also applies to the constrained case.

Theorem 4.3. There exist positive ε and δ such that if $\|(x^0,v^0) - (x^*,v^*)\| < \varepsilon$ and $\|(B^0)^{-1} - \nabla_{xx}^2 L(x^*,v^*)^{-1}\|_M < \delta$ and if $(x^+,v^+) = (x,v) - (\overline{B})^{-1}\nabla L(x,v)$ and $B^+ = \mathrm{BFGS}(B,\Delta x,y)$ where $y = \nabla_x L(x^+,v^+) - \nabla_x L(x,v^+)$, then B^+ is nonsingular and there exist positive $\alpha_1,\alpha_2,\alpha_3$ such that

$$(4.5)\quad \| (B^{+})^{-1} - \nabla^2_{xx}L(x^*,v^*)^{-1}\|_M \le \sqrt{1-\alpha_1\theta^2} + \alpha_2\sigma((x,v),(x^+,v^+))$$

$$\| B^{-1} - \nabla^2_{xx}L(x^*,v^*)^{-1}\|_M + \alpha_3\sigma((x,v)(x^+,v^+)),$$

where $\theta = \dfrac{\| [B^{-1} - \nabla^2_{xx}L(x^*,v^*)^{-1}]y\|}{\| B^{-1} - \nabla^2_{xx}L(x^*,v^*)^{-1}\| \;\; \| M^{-1}y\|}$ and

$$\sigma((x,v),(x^+,v^+)) = \max\{\| (x,v) - (x^*,v^*)\|, \|(x^+,v^+) - (x^*,v^*)\|\} .$$

Lemma 4.2 and theorem 4.3 can now be used to obtain the local results we need. Since the following theorem is similar to theorems in [13] and [29] the proof is omitted. A proof can be also found in Vardi [31].

<u>Theorem 4.4</u> There exist positive ε and δ such that if $\| (x^0,v^0) - (x^*,v^*)\| < \varepsilon$ and $\| (B^0)^{-1} - \nabla^2_{xx}L(x^*,v^*)^{-1}\| < \delta$ then for all k, $r^k = \| (\bar{B}^k)^{-1}\nabla L(x^k,v^k)\|$ is accepted. Furthermore, if $(x^{k+1},v^{k+1}) = (x^k,v^k) - (\bar{B}^k)^{-1}\nabla L(x^k,v^k)$ and $B^{k+1} = \mathrm{BFGS}(B^k,\Delta x^k,y^k)$, then $\| (x^{k+1},v^{k+1})-(x^*,v^*) \| < \varepsilon$ and $\| (B^{k+1})^{-1} - \nabla^2_{xx}L(x^*,v^*)^{-1}\| < 2\delta$ for all k. Finally, $\| (x^{k+1},v^{k+1}) - (x^*,v^*)\| \le t\| (x^k,v^k) - (x^*,v^*)\|$ for $0 < t < 1$ so that $\{(x^k,v^k)\}$ converges to (x^*,v^*) linearly.

<u>Theorem 4.5</u>. There exist positive ε and δ such that if $\| (x^0,v^0) - (x^*,v^*)\| < \varepsilon$ and $\| (B^0)^{-1} - \nabla^2_{xx}L(x^*,v^*)^{-1}\|_M < \delta$ and algorithm II is followed, then q-superlinear convergence in $\{\binom{x^k}{v^k}\}$ is achieved

Proof. Let ε and δ be those in Theorem 4.4. Then we obtain $\|(x^{k+1},v^{k+1}) - (x^*,v^*)\| \le t\|(x^k,v^k) - (x^*,v^*)\|$ for some $0 < t < 1$ and conclude that $\sum_{k=1}^{\infty} \|(x^k,v^k) - (x^*,v^*)\| < \infty$. The next step is to use (4.5) to show that

$$\lim_{k\to\infty} \frac{\|[(B^k)^{-1} - \nabla^2_{xx}L(x^*,v^*)^{-1}]y^k\|}{\|y^k\|} = 0,$$

(see [6]). We now use the inequality

$$\frac{\|[B^k - \nabla^2_{xx}L(x^*,v^*)]\Delta x^k\|}{\|\Delta x^k\|} \le \frac{\|I - B^k\nabla^2_{xx}L(x^*v^*)^{-1}\| \cdot \|y^k - \nabla^2_{xx}L(x^*,v^*)\Delta x^k\|}{\|\Delta x^k\|}$$

$$+ \frac{\|B^k\|\cdot\|[(B^k)^{-1} - \nabla^2_{xx}L(x^*,v^*)^{-1}]y^k\|}{\|y^k\|} \cdot \frac{\|y^k\|}{\|\Delta x^k\|},$$

to obtain

$$\lim_{k\to\infty} \frac{\|[B^k - \nabla^2_{xx}L(x^*,v^*)]\Delta x^k\|}{\|\Delta x^k\|} = 0. \tag{4.6}$$

This further implies

(4.7)

$$\lim_{k\to\infty} \frac{\left\|\left[\bar{B}^k - \nabla^2 L\begin{pmatrix} x^* \\ v^* \end{pmatrix}\right]\begin{pmatrix} \Delta x^k \\ \Delta v^k \end{pmatrix}\right\|}{\left\|\begin{matrix} \Delta x^k \\ \Delta v^k \end{matrix}\right\|} = \lim_{k\to\infty} \frac{\left\|\left[\begin{pmatrix} B^k & \nabla h(x^k) \\ \nabla h(x^k)^T & 0 \end{pmatrix} - \begin{pmatrix} \nabla^2_{xx}L(x^*,v^*) & \nabla h(x^*) \\ \nabla h(x^*)^T & 0 \end{pmatrix}\right]\begin{pmatrix} \Delta x^k \\ \Delta v^k \end{pmatrix}\right\|}{\left\|\begin{matrix} \Delta x^k \\ \Delta v^k \end{matrix}\right\|} = 0 .$$

Dennis and Moré [6] showed that this implies

$$\lim_{k\to\infty} \frac{\| (x^{k+1},v^{k+1}) - (x^*,v^*) \|}{\| (x^k,v^k) - (x^*,v^*) \|} = 0,$$

i.e., q-superlinear convergence in $\{(x^k,v^k)\}$ is achieved.

To conclude this section we discuss what happens when $\nabla^2_{xx}L(x^*,v^*)$ is not a positive definite matrix. In such case we cannot talk about $\| (B^0)^{-1} - \nabla^2_{xx}L(x^*,v^*)^{-1} \|_M < \delta$ as in theorem 4.3 because there may not be any positive definite B^0 which satisfies this inequality. Recall our assumption (2.5): $\forall\, z$ s.t. $\nabla h(x^*)^T z = 0$, $z^T \nabla^2_{xx} L(x^*,v^*) z > 0$; in terms of the Q-R decomposition (see (3.2), (3.3)) $Q^* \nabla h(x^*) \;^* = \binom{T^*}{0}$ and the partition $Q^* = \begin{bmatrix} \bar{Q}^* \\ \bar{\bar{Q}}^* \end{bmatrix}$, this inequality is equivalent to the statement:

$$\bar{\bar{Q}}^* \nabla^2_{xx} L(x^*,v^*) \bar{\bar{Q}}^{*T} \quad \text{is a positive definite matrix.}$$

Now the question becomes whether it is enough to assume that $(\bar{\bar{Q}}^0 B^0 \bar{\bar{Q}}^{0^T})^{-1}$ ($\bar{\bar{Q}}^0$ from the Q-R decomposition of $\nabla h(x^0)$) is close to $(\bar{\bar{Q}}^* \nabla^2_{xx} L(x^*,v^*) \bar{\bar{Q}}^{*T})^{-1}$ in order to get fast local convergences. Powell [24] made an important step towards answering the question positively. He showed that

$$\lim_{k\to\infty} \frac{\| \bar{\bar{Q}}^k [B^k - \nabla^2_{xx} L(x^*,v^*)] \bar{\bar{Q}}^{kT} \Delta x^k \|}{\| \Delta x^k \|} = 0 ,$$

implies two-step superlinear convergence

$$\lim_{k\to\infty} \frac{\| x^{k+1} - x^* \|}{\| x^{k-1} - x^* \|} = 0.$$

5. Implementation

Algorithm I is an iterative one. Steps 3-8 describe what happens in each specific iteration and will be discussed here in more detail. Since we confine ourselves in this chapter to one specific iteration, we omit the superscripts.

We use the Cholesky decomposition of B, $B = CC^T$ where C is a lower triangular matrix. All the computations from this point on will be done in terms of this matrix C. In each iteration instead of deriving B^{k+1} from B^k, we will derive C^{k+1} from C^k.

Initialization

x^0 does not have to be feasible. It is interesting to notice that if some of the constraints, say $h_1,\dots h_p, p \leq m$, are linear, then even if $h_i(x^0) \neq 0$ for some $i \leq i \leq p$, as soon as the radius of the trust region is large enough so that $\alpha(\lambda) = 1$ (say at the k^{th} iteration), then $h_i(x^k + \Delta x^k) = h_i(x^k) + \nabla h_i(x^k)^T \Delta x^k = 0$ for $i = 1,\dots,p$ and the linear constraints are satisfied for the rest of the iterative process. v^0 is set by the program to $[\nabla h(x^0)^T \nabla h(x^0)]^{-1}[h(x^0) - h(x^0)^T \nabla f(x^0)]$; it plays only a role in the initialization of μ_i's and in the initialization of C^0 which is assigned the value of the identity matrix multiplied by $0.1 \cdot \| \nabla_x L(x^0,v^0) \| / r^0$.

The Q-R Decomposition of $h(x)$ and Its Use

We now clarify what we do when $\nabla h(x)$ is not of full rank. Recall from (3.4) that Δx has to satisfy $\alpha(\lambda)h(x) + \nabla h(x)^T \Delta x = 0$. If $\nabla h(x)$ is not of full rank it means that there is dependency between the columns of $\nabla h(x)$, say $\Sigma d_i \nabla h_i(x) = 0$. If $\Sigma d_i h_i(x) = 0$ then the equations are redundant and at least one of them can be removed while if $\Sigma d_i h_i(x) \neq 0$ then the equations are inconsistent. We assume that this is a rare situation and when it happens in a specific iteration we remove redundant or inconsistent equations until the remaining set is independent. We record the number of times the system lacks full rank and if it happens more than m times we stop the iterative process.

Technically the removal of redundant or inconsistent equations is done in the following way: at each iteration we need a Q-R decomposition of $\nabla h(x)$, i.e., we have to find an orthogonal matrix, Q, a permutation matrix Π and a $\ell \times k$ matrix $\overline{T}$ with $\overline{t}_{ij} = 0$ for $i > j$, where $\ell = \text{rank}(\nabla h(x))$, such that $Q\nabla h(x)\Pi = \binom{\overline{T}}{0}$. (If $\ell = k$ then $\overline{T}$ is square and upper triangular.) The decomposition is obtained with the use of Householder transformations. At the end of the process, if fewer than k Householder transformations were used to obtain $\overline{T}$ it means that $\nabla h(x)$ was not of full rank. The equations that will be ignored are those that correspond to the remaining columns. (In the computer program we decide to stop the process if the norm of each remaining column is less than $10^{-14} \cdot |\overline{T}_{11}|$.)

Partition $\overline{T}$ into $\overline{T} = [T,S]$ where T is an $\ell \times \ell$ upper triangular matrix; also partition Q into $Q = \begin{bmatrix} \overline{Q} \\ \overline{\overline{Q}} \end{bmatrix}$ where $\overline{Q}$ has ℓ rows and let $\Pi = [\overline{\Pi},\overline{\overline{\Pi}}]$ where $\overline{\Pi}$ has ℓ columns. With these notations we solve the reduced systems $\alpha(\lambda)\overline{\Pi}^T h(x) + \overline{\Pi}^T \nabla h(x)^T \Delta x = 0$ to obtain

$$(5.1) \qquad \overline{Q}\Delta x = -\alpha(\lambda)\overline{h} \quad ,$$

where $\overline{h} = T^{-T}\overline{\Pi}^T h(x)$.

Step 3 of the Algorithm

Suppose we are in a new iteration at the point x with a corresponding Lagrange multiplier v and we have already observed $f(x)$, $h(x)$, $\nabla f(x)$ and $\nabla h(x)$, and have performed a Q-R decomposition of $\nabla h(x)$. Also available are a lower triangular C such that $CC^T = B$ is an approximation of the Hessian of the Lagrangian at (x,v) and r, the radius of the trust region. In order to compute $(\Delta x,\Delta v)$ let us first multiply (3.3) by $\bar{\bar{Q}}$ from the left to obtain

$$\bar{\bar{Q}}(B+\lambda I)(\bar{Q}^T\bar{Q}+\bar{\bar{Q}}^T\bar{\bar{Q}})\Delta x + \bar{\bar{Q}}\cdot\nabla h(x)\cdot\Delta v + \bar{\bar{Q}}\cdot\nabla_x L(x,v) = 0.$$

Observing that $\bar{\bar{Q}}\nabla h(x) = 0$, we then get

(5.2) $$\bar{\bar{Q}}\Delta x(\lambda) = -(\bar{\bar{Q}}B\bar{\bar{Q}}^T+\lambda I)^{-1}\bar{\bar{Q}}(\nabla f(x)+B\bar{Q}^T(\bar{Q}\Delta x)).$$

In order to take advantage of (5.2) we need a decomposiiton of the matrix $\bar{\bar{Q}}B\bar{\bar{Q}}^T$. This is best done in the following way: Define $M = \bar{\bar{Q}}C$ so that $MM^T = \bar{\bar{Q}}B\bar{\bar{Q}}^T$. Obtain a Q-R decomposition of M^T : $PM^T\Sigma = [{R \atop 0}]$ where P is an $n\times n$ orthogonal matrix, Σ an $(n-\ell)\times(n-\ell)$ permutation matrix and R an $(n-\ell)\times(n-\ell)$ upper triangular matrix (recall $\ell = \mathrm{rank}(\nabla h(x))$). Partition P into $P = \begin{bmatrix}\bar{P}\\ \bar{\bar{P}}\end{bmatrix}$ where P has $(n-\ell)$ rows. Thus we have

(5.3) $$(\bar{\bar{Q}}L)^T = M^T = \bar{P}^T R\Sigma^T$$

and

(5.4) $$\bar{\bar{Q}}B\bar{\bar{Q}}^T = \Sigma R^T R\Sigma^T; \quad \bar{\bar{Q}}B\bar{\bar{Q}}^T + \lambda I = \Sigma(R^T R+\lambda I)\Sigma^T.$$

At this point we have to decide whether we want to compute $\Delta x(0)$. We check whether the distance between x and $H \equiv \{x+\Delta x\colon \bar{\Pi}^T h(x)+\bar{\Pi}^T\nabla h(x)^T\Delta x = 0\}$

is not greater than r in which case $\Delta x(0)$ is obviously too long and we need $\lambda > 1$. The distance between x and H is $\|\bar{h}\| = \|T^{-T}\bar{\Pi}^T h\|$.

If $r > \|\bar{h}\|$ we compute $\Delta x(0)$ as follows:

$$\Delta x(0) = \bar{Q}^T(\bar{Q}\Delta x(0)) + \bar{\bar{Q}}^T(\bar{\bar{Q}}\Delta x(0)) \ ,$$

where $\bar{Q}\Delta x(0)$ is $-\bar{h}$ from (5.1). Defining

$$(5.5) \qquad b(\lambda) = C^{-1}\nabla f(x) + C^T\bar{Q}^T(\bar{Q}\Delta x(\lambda)) = C^{-1}\nabla f(x) - \alpha(\lambda)C^T\bar{Q}^T\bar{h} \ ,$$

we get from (5.2), (5.3) and (5.4)

$$(5.6) \qquad \Delta x(0) = -\bar{Q}^T\bar{h} - \bar{\bar{Q}}^T \Sigma R^{-1}\bar{P}b(0).$$

If now $\|\Delta x(0)\| < r$ we can find $v + \Delta v(0)$ (see below) and go to the next step. Otherwise we have to find $\lambda > 0$ such that $\|\Delta x(\lambda)\| = r$. We can get an upper bound on the value of λ in the following way:
From (5.1) and (5.2)

$$\Delta x(\lambda) = -\bar{\bar{Q}}^T(\bar{\bar{Q}}B\bar{\bar{Q}}^T + \lambda I)^{-1}\,\bar{\bar{Q}}(\nabla f(x) + B\bar{Q}^T(\bar{Q}\Delta x)) + \bar{Q}^T(\bar{Q}\Delta x).$$

Thus $r = \|\Delta x\| \leq \frac{1}{\lambda}\|\bar{\bar{Q}}\nabla f(x) + \alpha(\lambda)\bar{\bar{Q}}B\bar{Q}^T\bar{h}\| + \alpha(\lambda)\|\bar{h}\|$. This implies by straightforward analysis the following bound on λ

$$(5.7a) \qquad \lambda \leq u^0 \equiv \begin{cases} \dfrac{\|\bar{\bar{Q}}\nabla f(x) + \bar{\bar{Q}}B\bar{Q}^T\bar{h}\|}{r - \|\bar{h}\|} & \text{if } r \geq \|\bar{\bar{Q}}\nabla f(x) + \bar{\bar{Q}}B\bar{Q}^T\bar{h}\| + \|\bar{h}\| \\[2ex] \max\left\{\dfrac{\|\bar{\bar{Q}}\nabla f(x)\| + \|\bar{h}\| + \sqrt{(\|\bar{\bar{Q}}\nabla f(x)\| + \|\bar{h}\|)^2 + 4\|\bar{\bar{Q}}B\bar{Q}^T\bar{h}\| \cdot r}}{2r}, 1\right\} & \text{otherwise.} \end{cases}$$

A lower bound on λ can also be derived:

(5.7b) $$\lambda \geq \ell^0 = \begin{cases} 1 & \text{if } r \leq \|\bar{h}\| \\ 0 & \text{otherwise.} \end{cases}$$

Define now the function $\phi(\lambda) = \|\Delta x(\lambda)\|_2 - r$; we want to find a zero of this function. From theorem 3.2 we know that $\phi(\lambda)$ is continuous and decreases monotonically to $-r$. ϕ may not be always convex but it usually is. These characteristics of ϕ lead to an iterative process that was first suggested by Hebden [16] and Moré [19] for the nonlinear least square problem. In this process we set $\lambda^{j+1} = \lambda^j - \frac{\phi(\lambda^j) + r}{r} \frac{\phi(\lambda^j)}{\phi'(\lambda^j)}$ (λ^j is the j-th iterate) and check that $\lambda^{j+1} \in (\ell^{j+1}, u^{j+1})$ where ℓ^{j+1}, u^{j+1} are the best lower and upper bounds known for λ. In practice this method is quickly convergent and requires on average less than 2 λ-iterations per radius to obtain λ such that $|\phi(\lambda)| < 0.125 \cdot r$.

By using (5.3), (5.4) and (5.5) we can rewrite (5.2) as

$$\bar{\bar{Q}}\Delta x(\lambda) = -\Sigma(R^T R + \lambda I)^{-1} R^T \bar{P} b(\lambda)$$

or

(5.8) $$\begin{pmatrix} R \\ \lambda^{\frac{1}{2}} I \end{pmatrix}^T \begin{pmatrix} R \\ \lambda^{\frac{1}{2}} I \end{pmatrix} (\Sigma^T \bar{\bar{Q}} \Delta x(\lambda)) = \begin{pmatrix} R \\ \lambda^{\frac{1}{2}} I \end{pmatrix}^T \begin{pmatrix} -\bar{P} b(\lambda) \\ 0 \end{pmatrix} .$$

Thus we have a linear least squares problem and in order to solve it we have to obtain a Q-R decomposition of the matrix $\begin{pmatrix} R \\ \lambda^{\frac{1}{2}} I \end{pmatrix}$. This is done with the use of Givens transformations to get a $2(n-\ell) \times 2(n-\ell)$ orthogonal matrix W and an $(n-\ell)\times(n-\ell)$ upper triangular R such that $W\begin{pmatrix} R \\ \lambda^{\frac{1}{2}}I \end{pmatrix} = \begin{pmatrix} \tilde{R} \\ 0 \end{pmatrix}$. (We do not have to store W; the computations that have to be done with W are done while the decomposition is taking place.) Thus

$$R^TR + \lambda I = \tilde{R}^T\tilde{R}. \qquad (5.9)$$

Partition W into $\begin{pmatrix} \overline{W} \\ \overline{\overline{W}} \end{pmatrix}$ where $\overline{W}$ has $(n-\ell)$ rows. Thus

$$\begin{pmatrix} R \\ \lambda^{\frac{1}{2}}I \end{pmatrix} = \overline{\overline{W}}^T\tilde{R}\ . \qquad (5.10)$$

These decompositions enable us to compute $\Delta x(\lambda)$ and $\frac{\partial}{\partial\lambda}\|\Delta x(\lambda)\|$ and thus to obtain $\phi(\lambda) = \|\Delta x(\lambda)\| - r$ and $\phi'(\lambda) = \frac{1}{\|\Delta x(\lambda)\|}\cdot\frac{1}{2}\frac{\partial}{\partial\lambda}\|\Delta x(\lambda)\|^2$. For complete details see [31]. We also obtain

$$v + \Delta v = -\overline{\Pi}T^{-1}\overline{Q}[(B+\lambda I)\Delta x + \nabla f(x)].$$

This is the unique solution to (3.3) if $\nabla h(x)$ is of full rank in which case $\overline{\Pi}^T$ is square and invertible. When $\nabla h(x)$ is not of full rank we must decide on appropriate Lagrange multipliers for the constraints that have been ignored. The specific choice of $(v+\Delta v)$ above sets the multipliers for these constraints to zero which means that they will have no influence on B^+.

Steps 4-6 of the Algorithm

In step 4 of Algorithm I we use the penalty function $PL(x) = f(x) + \Sigma\mu_i|h_i(x)|$ to decide whether the step is acceptable.

We now describe how the μ_i's are set in $PL(x)$. We recall from theorem 3.3 that in order to guarantee descent of this penalty function we need

$$\mu_i > |\tilde{v}_i^j| = |[\nabla h(x^j)^T \nabla h(x^j)]^{-1}(h(x^j) - \nabla h(x^j)^T \nabla f(x^j))]_i| \ ,$$

for $i = 1,\ldots,m$; for all $j = 0,1,\ldots$.

In the program μ_i's are initialized to $\mu_i^0 = 2 \cdot |v_i^0|$. Then at each iteration we update the μ_i's by

$$(5.11) \qquad \mu_i^{j+1} = \begin{cases} 2 \cdot |\tilde{v}_i^j| & \text{if } 2 \cdot |\tilde{v}_i^j| \geq \mu_i^j \\ \tfrac{1}{2}(\mu_i^j + 2 \cdot |\tilde{v}_i^j|) & \text{if } 2 \cdot |\tilde{v}_i^j| < \mu_i^j \end{cases} .$$

As we commented in section 4, we modify the definition of $PL(x)$ so that when the x's are getting feasible, the regular Lagrangian function will replace the penalty function so that, as our local theorems show, q-superlinear convergence can result. This is done by gradually changing each of the μ_i's into a Lagrange multiplier when the point is in a neighborhood where the corresponding constraint is satisfied.

We have three tests that are designed to check three stopping criteria: the f convergence test, ∇L convergence test and x convergence test. In the program the user is asked to specify $\varepsilon_1 \geq 0$, $\varepsilon_2 \geq 0$, $\varepsilon_3 \geq 0$, $\varepsilon_4 \geq 0$, and funmin, a lower bound on the function.

f convergence test - stop if $f(x) - \text{funmin} < \varepsilon_1$ and $\|h(x)\| < \varepsilon_4$.

∇L convergence test - stop if $\|\nabla_x L(x,v)\| < \varepsilon_2$ and $\|h(x)\| < \varepsilon_4$.

x convergence test - stop if $\|\Delta x\| < \varepsilon_3(\|x\| + 1)$.

The test problems were run with $\varepsilon_1 = 10^{-8}$, $\varepsilon_2 = 10^{-5}$, $\varepsilon_3 = 10^{-10}$, $\varepsilon_4 = 10^{-6}$ on an IBM/370.

As for step 6, the program has a series of tests which is designed to compare the qudratic approximations of the Lagrangian function L with L itself., The new radius is set to $r^{k+1} = \begin{pmatrix} 2 \\ 1 \\ \frac{1}{2} \end{pmatrix} * \|\Delta x^k\|$ according to the fit. At times, when the raduction in the Lagrangian value is much better than expected we may recompute a step with $r^k = 2r^k$ before moving to the next iteration. Also if $PL(x^k + \Delta x^k) > PL(x^k)$ we will take $r^k = \frac{1}{2}r^k$ and recompute the step. For more details on assessing the quadratic model see Vardi [31].

Step 7 of the Algorithm

In (2.4) we gave the BFGS formula for updating the matrix B to obtain the matrix B^+. We now want to use the Cholesky decomposition of B, $B = CC^T$. Dennis and Schnabel [8] recommend the following method: obtain a Q-R decomposition of the matrix $(J^+)^T = C^T + v(y - Cv)^T/v^Tu$ where $v = (y^Ts/s^TBs)^{\frac{1}{2}}C^Ts$ by using Givens transformations and taking advantage of the fact that $(J^+)^T$ is a rank-one correction of an upper triangular matrix. Thus we get $(J^+)^T = Q^+(C^+)^T$. C^+ is then the Cholesky decomposition of the BFGS update of B, i.e., $B^+ = C^+C^{+T}$. Of course if $y^Ts \leq 0$, J^+ is not well defined and we just take $C^+ = C$.

The program also contains a subroutine that obtains a rough estimate of the condition number of B by checking the ratio between the largest diagonal element and the smallest diagonal element of C. If the ratio is too high the column that contains the low diagonal element is changed. In general the trust region model may prevent some of the numerical problems because the conditions number of the matrix $(\bar{\bar{Q}}B\bar{\bar{Q}}^T + \lambda I)$ as a function of λ is monotonically decreasing.

Testing the Program

The program has been tested extensively with test problems that appear in the literature. (See Himmelblau [17], Miele et al. [18], Solow [27] and Wright [32].) In order to check the global convergence, we added for each of these problems starting points that were much further from the known solution than the suggested starting points; convergence was always obtained.

For all problems we used $r^0 = 1$.

We give the results of the following problems:

Problem 1: $f(x) = (x_1 - x_2)^2 + (x_2 - x_3)^4$

$h_1(x) = x_1 + x_1x_2^2 + x_3^4 - 3.$

$f(x^*) = 0.$

Problem 2: $f(x) = 1000 - x_1^2 - 2x_2^2 - x_3^2 - x_1x_2 - x_1x_3$

$h_1(x) = x_1^2 + x_2^2 + x_3^2 - 25$

$h_2(x) = 8x_1 + 14x_2 + 7x_3 - 56.$

$f(x^*) = 961.71517.$

Problem 3: $f(x) = -(x_1 + x_2 + x_3 - 7)^3$

$h_1(x) = x_1^2 + x_2^2 + x_3^2 - 2$

$h_2(x) = x_2 - \exp(x_1).$

$f(x^*) = 117.0622.$

Problem 4: $f(x) = \exp(x_1x_2 - x_3^2)$

$h_1(x) = x_1^2 + x_3^4 - 2$

$h_2(x) = x_1x_2 - x_2^3 + x_3.$

$f(x^*) = 0.16550395.$

Problem 5: $f(x) = -x_1^2x_4 + (x_1-1)^4 + (x_2 - x_3)^4 + (x_3-1)^2$

$h_1(x) = x_1x_4^2 + \sin(x_4-x_3) - 4$

$h_2(x) = x_2^2 + x_3^2x_4^4 - 10.$

Two local solutions: $f(x^*) = -4.496926$; $f(x^*) = 1.9046409.$

Problem 6: $f(x) = x_1x_2x_3x_4x_5$

$h_1(x) = x_1^2 + x_2^2 + x_3^2 + x_1^2 + x_5^2 - 10$

$h_2(x) = x_2x_3 - 5x_4x_5$

$h_3(x) = x_1^3 + x_2^3 + 1.$

$f(x^*) = -2.9197004.$

Problem 7: $f(x) = \exp(x_1x_2x_3x_4x_5)$

constraints as in problem 6.

$f(x^*) = 0.053949848 = \exp(-2.9197004)$.

Problem 8: $f(x) = (x_1-1)^2 + (x_1-x_2)^2 + (x_2-x_3)^3 + (x_3-x_4)^4 + (x_4-x_5)^4$

$h_1(x) = x_1 + x_2^2 + x_3^3 - 2 - 3\sqrt{2}$

$h_2(x) = x_2 - x_3^2 + x_4 + 2 - 2\sqrt{2}$

$h_3(x) = x_1x_5 - 2$.

Five local solutions: $f(x^*) = 0.029310831$;

$f(x^*) = 27.871905$; $f(x^*) = 44.022072$;

$f(x^*) = 52.90258$; $f(x^*) = 607.03552$.

Problem 9: $f(x) = (x_1-1)^2 + (x_1-x_2)^2 + (x_3-1)^2 + (x_4-1)^4 + (x_5-1)^6$

$h_1(x) = x_1^2x_4 + \sin(x_4-x_5) - 2.8284$

$h_2(x) = x_2 + x_3^4x_4^2 - 9.4142$.

$f(x^*) = 0.24150237$.

Problem 10: $f(x) = (x_1-1)^2 + (x_1-x_2)^2 + (x_2-x_3)^4$

$h_1(x) = x_1(1+x_2^2) + x_3^4 - 8.2426$.

$f(x^*) = 0.032567769$.

Problem 11: $f(x) = (x_1-x_2)^2 + (x_2+x_3-2)^2 + (x_4-1)^2 + (x_5-1)^2$

$h_1(x) = x_1 + 3x_2$

$h_2(x) = x_3 + x_4 - 2x_5$

$h_3(x) = x_2 - x_5$.

$f(x^*) = 4.0930233$.

Problem 12: $f(x) = 4x_1^2 + 2x_2^2 + 2x_3^2 - 33x_1 + 16x_2 - 24x_3$

$h_1(x) = 2x_2^2 + 3x_1 - 7$

$h_2(x) = x_3^2 + 4x_1 - 11$

$f(x^*) = -99.555041$.

TEST RESULTS

Problem number	Starting point	No. of Function eval.	No. of Gradient eval.
1	(2.4,.5,0)	28	24
1	(10,-10,10)	78	50
2	(-5,-10,5)	18	11
2	(10,10,10)	20	14
2	(50,50,50)	23	15
3	(0,1,1)	11	8
3	(-10,10,10)	34	21
4	(-1,1,1)	9	8
4	(-10,10,10)	22	15
5	(3.159,3.162,0,1)	19	16
5	(10,10,10,10)	47	35
6	(-1,1.5,2,-1,-2)	18	10
6	(-10,10,10,-10,-10)	25	11
7	(-2,2,2,-1,-1)	8	8
7	(-1,-1,-1,-1,-1)	14	10
8	(-1,3,-0.5,-2,-3)	16	13
8	(-1,2,1,-2,-2)	15	12
8	(1,1,1,1,1)	13	11
8	(2,2,2,2,2)	13	10
8	(10,10,10,10,10)	25	21
8	(-2,-2,-2,-2,-2)	45	22
9	(2,2,2,2,2)	16	14
9	(10,10,10,10,10)	91	64
10	(1.5,1.5,1.5)	13	11
10	(10,10,10)	20	15
11	(2,2,2,2,2)	13	9
11	(10,10,10,10,10)	16	11
12	(4,-3,4)	11	10
12	(10,-10,10)	15	11

Acknowledgment

I would like to express my sincere appreciation to M. J. Todd and J. E. Dennis for their guidance and helpful suggestions which have greatly contributed to this work.

References

[1] M. Avriel, *Nonlinear Programming: Analysis and Methods*, Prentice-Hall, Inc, 1976.

[2] C. G. Broyden, *The convergence of single rank quasi-Newton methods*, Math. Comput., 24 (1970), pp. 365-382.

[3] C. G. Broyden, J. E. Dennis, and J. J. Moré, *On the local and superlinear convergence of quasi-Newton methods*, J. Inst. Math. Appl., 12 (1973), pp. 223-246.

[4] R. M. Chamberlain, C. Lemarechal, H. C. Pedersen, and M. J. D. Powell, *The watchdog technique for forcing convergence in algorithms for constrained optimization*, presented at the Tenth International Symposium on Mathematical Programming, Montreal (1980).

[5] A. C. Cline, C. B. Moler, G. W. Stewart, and J. H. Wilkinson, *An estimate for the condition number of a matrix*, SIAM J. Numer. Anal., 16 (1979), pp. 368-375.

[6] J. E. Dennis and J. J. Moré, *A characterization of superlinear convergence and its applications to quasi-Newton methods*, Math. Comput. (1974), pp. 549-560.

[7] J. E. Dennis and J. J. Moré, *Quasi-Newton methods, motivation and theory*, SIAM Review, 19 (1977), pp. 46-89.

[8] J. E. Dennis and R. B. Schnabel, *Quasi-Newton methods for unconstrained nonlinear problems*. Manuscript (1980) in preparation.

[9] J. J. Dongarra, C. B. Moler, J. R. Bunch, and G. W. Stewart, *LINPACK Users Guide*; SIAM.

[10] R. Fletcher, *A new approach to variable metric algorithms*, Comput. J., 13 (1970), pp. 317-322.

[11] P. E. Gill, W. Murray, and R. A. Pitfield, *The implementation of two revised quasi-Newton algorithms for unconstrained optimization,* Nat. Phys. Lab. NAC II (1972).

[12] D. Goldfarb, *A family of variable metric methods derived by variational means,* Math. Comput., 24 (1970), pp. 23-26.

[13] S.-P. Han, *Superlinearly convergent variable metric algorithms for general nonlinear programming problems,* Math. Programming, 11 (1976), pp. 263-282.

[14] S.-P. Han, *A globally convergent method for nonlinear programming,* J. Optimization Theory Appl., 22 (1977), pp. 297-309.

[15] S.-P. Han, *A hybrid method for nonlinear programming,* TR 78-331, Department of Computer Science, Cornell University (1978).

[16] M. D. Hebden, *An algorithm for minimization using exact second derivatives,* A.E.R.E., Harwell, T.P. 515 (1973).

[17] D. M. Himmelblau, *Applied Nonlinear Programming,* McGraw-Hill, 1972.

[18] A. Miele, P. E. Moseley, A. V. Levy, and G. M. Coggins, J. Optimization Theory Appl., 10 (1972), pp. 1-33.

[19] J. J. Moré, *The Levenberg Marquardt Algorithm: Implementation and theory,* Numerical Analysis, Lecture Notes in Math 630, Editor, Watson, Springer-Verlag, 1978.

[20] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equation in Several Variables,* Academic Press, 1970.

[21] M. J. D. Powell, *Convergence properties of a class of minimization algorithms,* NLP2, Editors, Meyer and Robinson, Academic Press (1975).

[22] M. J. D. Powell, *Algorithms for nonlinear constraints that use Lagrangian functions,* Presented at the Ninth International Symposium on Math. Programming, Budapest, 1976.

[23] M. J. D. Powell, *Variable metric methods for constrained optimization,* Presented at the Third International Symposium on Computing Methods in Applied Sciences and Engineering, Paris, 1977.

[24] M. J. D. Powell, *The convergence of variable metric methods for nonlinearly constrained optimization calculations*, NLP3, Editors, Meyer and Robinson, Academic Press, 1978.

[25] M. J. D. Powell, *A fast algorithm for nonlinearly constrained optimization calculations*, Numerical Analysis, Lecture Notes in Mathematics 630, Editor, Watson, Springer-Verlag, 1978, pp. 144-157.

[26] D. F. Shanno, *Conditioning of quasi-Newton methods for function minimization,* Math. Comput., 34 (1970), pp. 647-656.

[27] D. Solow, *Decomposition in Fixed Point Computation,* Ph.D. Thesis, Stanford University, 1977.

[28] G. W. Stewart, *Introduction to Matrix Computation,* Academic Press, 1973.

[29] R. A. Tapia, *Diagonalized multiplier methods and quasi-Newton methods for constrained optimization,* J. Optimization Theory & Appl., 22 (1977), pp. 135-194.

[30] R. A. Tapia, *Quasi-Newton methods for equality contrained optimization: equivalence of existing methods and a new implementation,* presented at the Nonlinear Programming Symposium 3, Madison, WI, 1977.

[31] A. Vardi, *Trust region strategy for unconstrained and contriand minimization*, Ph.D. Thesis, School of Operations Research and Industrial Engineering, Cornell University, 1980.

[32] M. Wright, *Numerical methods for nonlinearly constrained optimization,* SLAC Report No. 193, Stanford University, 1976.